An efficient optimizer for simple point process models

Ahmed Gamal Eldin, Guillaume Charpiat, Xavier Descombes, Josiane Zerubia

An efficient optimizer for simple point process models

Ahmed Gamal-Eldin[a] [b], Guillaume Charpiat[a], Xavier Descombes[a] and Josiane Zerubia[a]

[a]INRIA Sophia Antipolis, 2004, route des Lucioles, 06902 Sophia Antipolis, France;
[b]INRIA Rhone Alpes, 655 Avenue de l'Europe, 38330 Montbonnot, France

ABSTRACT

In this paper we discuss the main characteristics (that we consider to be essential) for the design of an efficient optimizer in the context of highly non-convex functions. We consider a specific model known as Marked Point Process (MPP). Given that the probability density is multimodal, and given the size of the configuration space, an exploration phase is essential at the beginning of the algorithm. Next, the fine details of the density function should be discovered. We propose efficient kernels to efficiently explore the different modes of the density, and other kernels to discover the details of each mode. We study the algorithm theoretically to express convergence speeds and to select its best parameters. We also present a simple and generic method to parallelize the optimization of a specific class of MPP models. We validate our ideas first on synthetic data of configurations of different sizes to prove the efficiency of the proposed kernels. Finally we present results on three different applications.

Keywords: Point process, multiple birth and cut, non-convex optimization, multiple object detection

1. INTRODUCTION

In the last decade, multiple object detection has evolved in both models and optimizers. We find two different mainstreams from the aspect of object modeling. *Inverse methods* follow a bottom-up approach: they construct the objects from primitives, like corners, edges, feature points or segmented object parts, by a rule based model or a previously trained classifier.[1] Recognition with these solutions can be notably quick due to their sequential workflow, however, their performance may drastically decrease if the features cannot be reliably detected. On the other hand, *direct methods* assigning a fitness value to each possible configuration, and an optimization process attempts to find the configuration with the highest confidence. This way, flexible object appearance models can be adopted, and it is also straightforward to incorporate prior shape information and object interactions into the model.

However, the optimization process can be a bottleneck for many sophisticated models. The richer the model is, the more complex its optimization becomes. This phenomenon makes most of the research focus on very simple models that can easily be optimized, even if these models can be over-simplistic for some applications. Markov random field (MRF) models are life examples that suffer this type of problem. While higher interactions with sophisticated potential terms exhibit more general modeling capabilities, their optimization is very hard. Consequently, most of the popularly used models are simple models that can be efficiently and rapidly optimized. But these models can be oversimplistic for some applications. Considering MRFs, the most widely used class is of first order with limited set of potential terms, because this class can be efficiently optimized with graph cut algorithms.[2,3] More sophisticated models require stochastic samplers such as Gibbs and Monte Carlo Markov Chain (MCMC) algorithms. In this work we consider a multiple object detection framework known as Marked point process. This framework has shown a great potential in remote sensing applications, such as building detection, tree detection, road detection, and many others. The strength of this model lies in its capacity to incorporate a wide variety of data terms and also a wide range of prior information. Data terms for object detection can range from simple correlation to using an SVM classifier trained on SIFT features for specific object category. The prior term also can range from simple non-overlapping condition to prior information about minimal distances and angles between objects.

Further author information, E-mail: firstname.lastname@inria.fr

Here the optimization issue plays a crucial role due to the high dimension of the configuration space. Until very recently, the only existing methods to optimize MPP models were fully stochastic. The initial optimizer is Birth-and-Death,[4] then came the Reversible Jump Markov Chain Monte Carlo (RJMCMC)[5] and recently came the Multiple-Birth-and-Death (MBD).[6] While each of these optimizers represents on its own a major advance in both the optimization world and in the application world, these algorithms are fully stochastic and are embedded in the simulated annealing scheme. Although those algorithms may be able to optimize very complex models with huge configuration space, they are very slow in practice.

A recent work[7] was the first to show the possibility of optimizing such models (MPP) with a mixed semi-deterministic optimizer using graph-cut. This new algorithm is known as Multiple Birth and Cut (MBC). This method holds two parts, the stochastic part to explore a very large configuration space, and the deterministic part which makes the optimal selection between an existing configuration and a newly proposed one.

In this paper, after a presentation of our Point Process Model, we propose methods to boost the efficiency of the MBC optimizer. First, we propose a simpler and faster way to speed up the birth step of this algorithm than in.[8] Next, we introduce concepts from the stochastic MCMC optimizer, in a simpler way that will again improve the speed of the optimizer. Then in section 5, we study theoretically the complexity to express convergence properties, and present how this algorithm scales on a synthetic problem, compared to the existing algorithms. Finally, in section 6, we show some of the many possible applications to which this type of modeling can be applied.

2. POINT PROCESS MODEL

We consider that point process models in image processing can be divided into two classes. The first class covers models with simple interactions. Examples for this class could deal with trees, flamingos, cells, and many others. The second class is appropriate for modeling scenes of objects with complex interactions (priors), such as roads. For example, road segments have to be connected, and the intersection angle can not be less than 5°. In this paper we focus on the first class. In the following, we briefly introduce point process models.

We consider a point process X living on a space K. K is the image and is defined by $K = [0, I_{max}] \times [0, J_{max}]$, it is a closed connected subset of $\mathbb{R}^2$. The point process X is a mapping from a probability space $(\Omega, \mathcal{A}, \mathcal{P})$ to the set of configurations of points on K. The points on their own represent the objects' centers, while the marks hold the geometrical parameters of these objects. Each object ω_i consists of a point x_i and a mark m_i associated to this point; let M is the mark space. The configuration space is then defined as:

$$\Omega = \bigcup_{n=0}^{\infty} \Omega_n, \quad \Omega_n = \{\{\omega_1, \ldots, \omega_n\} \subset K \times M\} \,, \tag{1}$$

where Ω_n is the subset of configurations containing exactly n objects. The number n of objects to be detected is a random variable which is unknown.

The density of the process is defined by a Gibbs density on the configuration space. The Gibbs model provides a powerful tool for modeling a scene of objects and interactions between them. The energy is written as the sum of potentials over interacting objects (cliques). The density is therefore defined by:

$$f(\omega) = \frac{1}{Z} exp[-U(\omega)] = \frac{1}{Z} exp[-(U_d(\omega) + \gamma_p U_p(\omega))] \,, \tag{2}$$

where U_p is the prior energy which takes into account the interactions between geometric objects, U_d is the data energy to fit the configuration to the image, Z is the partition function and γ_p is the weight of the prior term.

2.1 Data Term

As we mentioned in the introduction, the MPP framework does not impose any limitations on the selected method for evaluating the fitness of a proposed object. Given the independence of the data term of each object, the data term energy of a configuration ω is given by $U_d(\omega) = \sum_{\omega_i \in \omega} u_d(\omega_i)$ The term $u_d(\omega_i)$ evaluates from a data point of view the relevance of the proposed object ω_i. The object contains information on both its location and its shape.

2.2 Prior Term

The capacity to incorporate prior information (for regularization) is one of the major advantages of MPP models. Since in this work we only consider the first class of MPP models, we have a simple prior term which consists in penalizing overlapping objects, as objects in the selected applications in the real world do not overlap. Let $\mathcal{A}(\omega_i, \omega_j) \in [0, 1]$ be the overlapping coefficient between two objects. As required by the MBC algorithm for the optimization of the first class of MPP models, the interaction cost should be zero or infinity. The prior energy of a local configuration $\{\omega_i, \omega_j\}$, where $\omega_i \sim \omega_j$ ($\sim$ indicates neighbor objects) is given by:

$$u_p(\omega_i, \omega_j) = \begin{cases} 0 & \text{if } \mathcal{A}(\omega_i, \omega_j) < 0.1 \\ \infty & \text{if } \mathcal{A}(\omega_i, \omega_j) \geq 0.1, \end{cases} \tag{3}$$

which means that we tolerate small overlapping up to 10%, otherwise it costs infinity. The total prior configuration energy is given by $U_p(\omega) = \sum_{(\omega_i \sim \omega_j) \in \omega} u_p(\omega_i, \omega_j)$.

3. OPTIMIZATION

Now that we have defined the density by an energy function $U(.)$, we want to compute the Maximum Likelihood (ML) estimate with respect to the density f. The ML estimate corresponds to the configuration that matches best our model and the input image. The estimate of the global mode of a density is given by:

$$\hat{\omega} = \underset{\omega \in \Omega}{\operatorname{argmax}}\, f(\omega) = \underset{\omega \in \Omega}{\operatorname{argmin}}\, U(\omega)$$

Finding the minimum (or maximum) for this highly non-convex function (equation 2) with many local minima requires a global optimization method.

3.1 What makes a good optimizer?

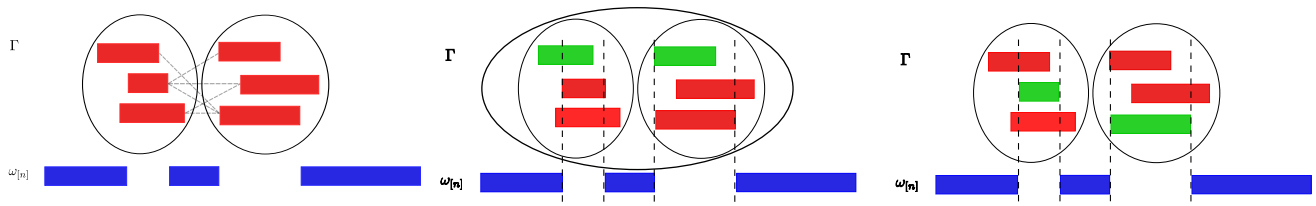
In this section we will discuss what we consider to be the main characteristics of optimizers[9, 10] for non-convex functions in the context of MPP models.

1. Global versus local proposals: Local proposal* means exploring the *fine details of the modes*, while global proposal means exploring the *different modes*. For a multimodal distribution, and with only local proposals, based on the initialization (starting point), only one mode will be discovered, while using only global proposals will not capture the details the modes (it will be very slow to capture the detailes). For either simulation or optimization, given that the considered density is multimodal, both these properties are required.

2. Simple versus multiple perturbations: The perturbations are simple or multiple based on how the configuration changes between two iterations. If only *one object* is modified (changed, removed or added), then it is a simple perturbation. If *many objects* are modified per iteration, then it is considered as a multiple perturbation.

Holding those characteristics is essential to make a fast optimizer. It requires global and local perturbation kernels, and also to be able to make multiple perturbations (simple perturbation is implicite).

*Proposal, also named move or perturbation, means making a modification to the current configuration $\omega_{[n]}$ (at iteration n)



(a) (b) (c)

Figure 1: (a) A synthetic 1D problem. Current configuration $\omega_{[n]}$ shown in blue. Candidate objects shown in red. Interaction terms are presented with dotted gray lines. (b) Best candidates globally are shown in green. (c) Best candidates locally are shown in green.

3.2 Kernels Evolution in Point Process models

- In a multiple object detection problem, the dimensionality of a set is unknown in advance, as it is the number of objects to be detected. This means that only samplers that can make dimensional jump can be used. This was only possible after the development of a *birth-death* kernel by Geyer et al.[4]
- MCMC is a flexible sampler, it can incorporate different kernels that are problem specific, but it can not make a dimensional jump. By integrating the *birth-death* kernel into MCMC, Green obtained a new sampler that can also make dimensional jump. This sampler is known as RJMCMC.
- Using the previous samplers, we can only make simple perturbations. The problem is that in practice those samplers are very slow. Due to the work of Descombes et al.,[6] it became possible to make multiple perturbations. They developed a *multiple-birth-death* kernel, and this is global multiple perturbations kernel.
- All the above samplers are fully stochastic embedded in a simulated annealing scheme. It is only very recently that a semi-deterministic optimizer was introduced by Gamal-Eldin et al.[7] This optimizer is known as Multiple Birth and Cut (MBC). It also has a *multiple-birth-death* kernel for global multiple perturbations.

4. NEW MBC ALGORITHM

We start from the MBC algorithm presented in.[7] We propose a more efficient *multiple-birth-death* kernel than the one proposed in.[8] The proposed kernel makes more efficient global perturbations, and explore more efficiently the configuration space, while being very simple. We next show how local perturbation kernels can be integrated in the MBC algorithm to make it more efficient.

4.1 New selection method

We start by analyzing the behavior of the selection method based on belief propagation presented in.[8] We consider a 1D problem as illustrated in figure 1(a). Let $\omega_{[n]}$ be the current configuration, presented in blue in figure 1. Assume that the two empty spots should contain objects. Over each of the two empty spots a packet of objects is proposed. The packets are presented in red in the figure. The aim is to select the best candidate object inside each packet X_i. There exist some interactions (presented by the gray dotted lines) between some objects of different packets. Each object has a data term (cost) and a prior term (interaction cost). Let the set of proposed packets be defined by $\Gamma = \{X_0, X_1, \ldots, X_n\}$. The belief propagation algorithm makes the best selection from Γ (global optimal), based on the data terms and the prior terms. The algorithm ends by selecting the two candidate objects $\{\omega_1^1, \omega_2^1\}$, shown in green in figure 1(b).

While the MAP estimate using belief propagation gave the optimal selection from Γ, those selected objects were not necessarily optimal from the point of view of the current configuration $\omega_{[n]}$. As figure 1(c) illustrates, what was optimal for the current configuration $\omega_{[n]}$ was $\{\omega_1^2, \omega_2^3\}$ and not $\{\omega_1^1, \omega_2^1\}$. The difference is, $\{\omega_1^2, \omega_2^3\}$ is locally optimal, while $\{\omega_1^1, \omega_2^1\}$ is optimal over Γ.

The aim is to minimize the total energy. If we consider only the data terms, $u_d(\omega_1^1)+u_d(\omega_2^1) < u_d(\omega_1^2)+u_d(\omega_2^3)$, so ω_1^1 and ω_2^1 are *perfect* candidates, while ω_1^2 and ω_2^3 are *good* candidates. If we consider the total energy, prior term and data term, $u_d(\omega_1^1) + u_d(\omega_2^1) + u_p(\omega_1^1, \omega_2^1) > u_d(\omega_1^2) + u_d(\omega_2^3)$.

We propose to make the selection only locally, and remove objects that violate the *non-overlapping* condition. We *re-propose* new objects instead of the rejected ones. It costs much more to get a perfect candidate than to get a good one. So keeping one of the two perfect candidates and re-proposing instead of the other is more efficient. In real applications, the percentage of rejected objects ranges between 5% and 10%. We keep iterating the algorithm until re-proposing this small percentage. The main reasons for being faster are:

1. It takes more time to propose an excellent candidate than a good one.
2. Let the size of Γ be $m \times n$, where m is the number of packets and n is the number of objects per packet. The complexity of belief propagation is $O(mn^2)$, while for local selection it is $O(mn)$.
3. Proposed configurations can be very dense, and they can form loops.
4. The belief propagation algorithm require extra computational cost to ensure that the proposed packets do not form any loop.

4.2 Local Perturbations Kernels

The existing multiple perturbation optimizers hold a single kernel, a *birth* kernel that proposes new objects. What is missed here is making local proposals.

Consider object ω_i to be the best candidate we obtained from the birth kernel to detect object $\mathbf{o}_i$. Object $\mathbf{o}_i$ has a position x_i and a set of marks m_i. If we consider the ellipse model in $\mathbb{R}^2$, the set of parameters of ω_i is $\Theta = \{x_i, y_i, a_i, b_i, \rho_i\}$. Given that object ω_i is a good candidate for the detection of object $\mathbf{o}_i$, this means that $\Theta_{\mathbf{o}_i}$ is close to Θ_{ω_i}. It is much easier to go from $\Theta_{\mathbf{o}_i}$ to Θ_{ω_i} (or closer) than to propose a new object that becomes a better candidate for object $\mathbf{o}_i$. This can be obtained via perturbation kernels.

Let us consider four perturbation kernels. One for perturbing the object position, a second to perturb the major axis, a third for the minor axis and a fourth one for the angle. From the current configuration $\omega_{[n]}$ we randomly select a subset ω'' of objects. On each object $\omega_i \in \omega''$ we apply a perturbation kernel, $\omega'' \xrightarrow{perturbations} \omega'$. More than one kernel can be applied to each object. Kernels are selected randomly. Giving different weights to the kernels may be beneficial. This type of perturbation is usually referred to as *local random walk*. Those kernels were originally proposed in the context of RJMCMC sampler.[11] While algorithm is not and MCMC algorithm, but we were able to take advantage of the many kernels idea. The new MBC algorithm is summarized in algorithm 1.

Algorithm 1 New Multiple Birth and Cut

$n \leftarrow 0$, $R \leftarrow$ *const1*
generate ω' , $\omega_{[0]} \leftarrow \omega'$
repeat
  Birth:
  Sample $u \sim \mathcal{U}_{(0,1)}$
  Based on u, select a kernel randomly (including the birth kernel)
  $\omega' \leftarrow$ Apply selected kernel
  $\omega \leftarrow \omega_{[n]} \cup \omega'$
  Cut: $\omega_{[n+1]} \leftarrow \text{Cut}(\omega_{[n]} \cup \omega')$ (optimize with graph cuts)
until converged

Global proposals are mostly needed at the starting of the algorithm. We force for the first few iterations to only use of the *birth* kernel. Thereafter we alternate between all the kernels. The proper selection of the models parameters and dynamic kernels selection are based on the following analysis.

5. ALGORITHM ANALYSIS

5.1 Theoretical analysis

5.1.1 Notations.

Algorithm 1 alternates randomly two kinds of steps: global exploration, with probability p_G, and local exploration, with probability $p_L = 1 - p_G$. A global exploration consists in picking randomly $\alpha_G N$ locations in the image (using the birth map, where N is a rough estimate of the number of objects to be found, and $\alpha_G \in [0, 1]$ a proportion) and in testing n_G random marks at each of these locations. A local exploration consists in picking randomly $\alpha_L N$ already detected objects, and in testing n_L random small variations of their marks (*i.e.* exploring direct neighbors in the discrete space of marks M of angle, axes' lengths and precise location).

5.1.2 Preliminary step.

Before applying Algorithm 1, a series of successive global explorations is performed, in order to ensure that most of the image domain K has been covered and that the number of objects detected (even if with wrong marks) is close to N. The average proportion of the image still uncovered after k such steps, *i.e.* the average proportion of regions with area $\frac{|K|}{N}$ not visited yet, is $(1 - \alpha_G)^k$. For $\alpha_G = 60\%$ *e.g.*, $k = 14$ steps are sufficient to ensure that 99.9% of the image has been explored.

5.1.3 Global vs. local explorations.

Now, at step s of Algorithm 1, the average number of random tests already performed in the neighborhood (of area $\frac{|K|}{N}$) of any image point x is $v_G s$ with $v_G := \alpha_G p_G n_G$, and the average number of local explorations around any detected object is similarly $v_L s$ with $v_L = \alpha_L p_L n_L$. Thus the ratio of local tests over global tests at any location is $\frac{v_L}{v_G} = \frac{1-p_G}{p_G} \frac{\alpha_L n_L}{\alpha_G n_G}$. However, the probability that a global test is successful decreases with time: by construction it is at most the probability that a random try in M performs better than the previous $v_G s - 1$ ones, which is $\frac{1}{v_G s}$. Consequently, the expected number of local explorations between two big moves in M increases with time and is $\frac{v_L}{{v_G}^2 s}$, which means that local minima are more thoroughly explored with time.

5.1.4 Time complexity and Percolation.

We will now study the time spent at each step, in order to optimize the algorithm *w.r.t.* parameters v and α. The cost of any exploration step, global or local, is of the form $\alpha n N$, plus the cost of applying graph cut to a graph of approx. $2\alpha N$ nodes. The expected complexity of graph cut is much lower than its worst-case complexity, and is usually regarded as between linear and quadratic in the size of the graph, depending on problems. The graph here however is generally not connected and thus the complexity depends on the size of its connected components (the smaller, the lower). Since the graph is made of randomly selected objects with density α, we are precisely interested into *percolation* properties, to obtain the typical number and size of connected components.[12] The critical site-percolation density α_c, above which arbitrarily big components arise, depends on the neighborhood size and on the type of lattice the graph is extracted from. In our application case, the number of neighbors is at most 6, and one can lay an hexagonal lattice on the image, with step size the typical object size. We could also consider the Voronoï diagram associated to the objects already detected, which is relatively similar. In this hexagonal setting, α_c is known to be approx. 0.697... . Since α has to satisfy $\alpha < \alpha_c$, a very cautious choice is to set $\alpha \leqslant 60\%$. For such an α, the phase is *subcritical* and the probability of connected regions is known to decrease exponentially with their area A, as $e^{-A/R}$, where R depends on α (smoothly for α far from α_c). The typical region size is then R, and, by computing $\left(\int_A A^2 e^{-A/R} dA\right) / \left(\int_A e^{-A/R} dA\right)$, we obtain that the average graph cut cost (supposing quadratic complexity) is proportional to R^2. One can bound R by $R_M = R(0.6)$ if we restrict $\alpha \leqslant 60\%$. Then the average total graph cut cost at step s, which involves $2\alpha N$ objects in about $\frac{\alpha N}{R}$ connected components, is bounded by $\alpha N R_M$. Consequently, the complexity of step s is bounded by $C_S = \alpha N(n + R_M)$. Note that this bound does not depend on s. The expected clock time at the end of step s is thus
$t = (p_G C_S(G) + p_L C_S(L))\, s = (v_G + v_L + R_M[p_G \alpha_G + (1 - p_G)\alpha_L])\, N s$

which has to be compared to $v_G s$ and $v_L s$, the average number of explorations in the neighborhood of any image point.

5.1.5 Estimated time and precision when in an attraction basin.

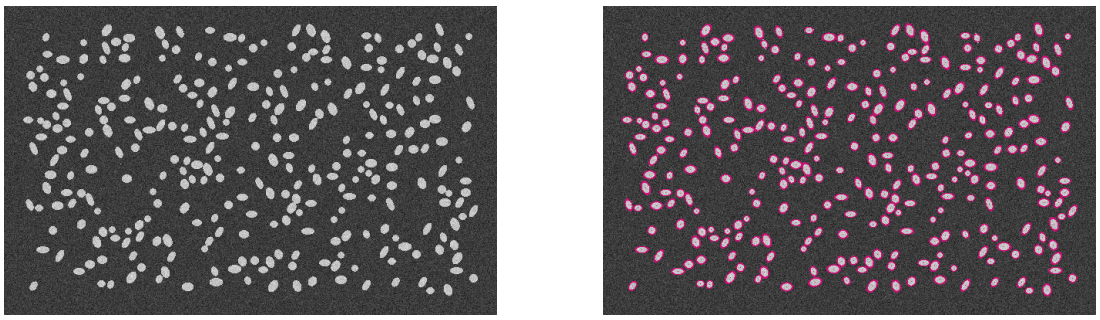
Let us consider the case of a locally simple problem, in the sense that objects are located away enough from each other, in order to prevent any mutual interaction. Under this hypothesis, any suitable data energy will satisfy that each individual potential admits a unique minimum, *i.e.* that the attraction basin of any ground-truth object includes all possible marks located in the neighborhood of this object. We can then estimate the time needed to find the global minimum in the mark space M with the method above. After x random global explorations in an image region of area $\frac{|K|}{N}$, the minimum is at distance $\frac{|M|}{x}$ on average and thus then reachable in $\frac{|M|}{x}$ local explorations. Minimizing $x + \frac{|M|}{x}$ leads to $x = \sqrt{|M|}$. Thus we aim at minimizing t such that $v_G s \geqslant \sqrt{|M|}$ and $v_L s \geqslant \sqrt{|M|}$, w.r.t. all parameters. This leads to minimizing $t = \left[(1 + \frac{R_M}{n_G})(v_G s) + (1 + \frac{R_M}{n_L})(v_L s)\right] N$, which is solved by $v_G = v_L$, and by high values of $n_G = n_L >> R_M$ to reduce the relative cost of graph cuts (we choose $n = 10$ in practice since R_M observed is already low). The corresponding average time to find the optimal mark for an object is $2\sqrt{|M|}N$. More precisely, the probability that the optimal mark is not found for this object after time t is $p_N = \frac{|M|!}{(M - t/N)!\, M^{t/N}} \approx e^{-\frac{1}{|M|}\left(\frac{t}{N}\right)^2}$ using Stirling formula. Hence, the probability to have already found it at time $\sqrt{|M|}N$ is about 0.63, and at time $2\sqrt{|M|}N$ about $e^{-4} \approx 0.98$. Furthermore, the probability p to have found all N objects at time $t = \gamma\sqrt{|M|}N$ is $(1 - e^{-\gamma^2})^N \approx e^{-N exp(-\gamma^2)}$; conversely the time needed to reach a given probability p is $t = \gamma(p)N\sqrt{|M|}$ with $\gamma(p) = \sqrt{-\ln \frac{-\ln p}{N}}$. Furthermore, the expectancy of the proportion of the objects not found yet after time $t = \gamma\sqrt{|M|}N$ is $e^{-\gamma^2}$ (in the case of a spatially uniform birth map). Thus a reasonable total time to spend in the locally convex case to expect **all but one** objects and their optimal marks is then $\sqrt{|M|}N\sqrt{\ln N}$.

5.1.6 Fixing local minima.

The results above remain valid for any problem provided objects are correctly localized from the first steps. However, it may happen, in the case of an object category with very-varied sizes, that two contiguous small objects are proposed instead of a big one. In such a case, one needs a global exploration at the correct location with reasonably good marks. The required quality of the birth depends on the relative energy cost of the wrong couple of small objects w.r.t. the one of the correct detection. Given a lower bound $B > 0$ on this difference (based on misplaced edges *e.g.*), it is sufficient that a global exploration finds a mark with energy less than B + the optimal one. Let p_{GM} be the proportion of such marks within the local mark space M. The average time to find one of them is $\frac{1}{2p_{GM}}N$. With similar techniques as before, the time required to fix, with probability p, simultaneously k such local minima occurring in the image, is $\ln(1 - p^{1/k})\frac{1}{2p_{GM}}N$. Note that another, faster possibility would be to add a merge-and-split kernel dedicated to this kind of minima.

5.1.7 Semi-local minima.

Depending on the energy minimized to detect objects, and in the case of overlapping neighborhoods of objects, bad initial explorations may result in wrong detections half-way between two real objects. Such wrong detections are fixed quickly by Algorithm 1, when a global exploration is performed at one of these locations, which happens as frequently as $\frac{2p_G}{\alpha}$. What is more problematic is when such wrong detections form chains of misplaced objects, each one preventing its neighbors from evolving to its right place. If the chain is a tree of diameter d, then the expected time to remove it is only $d\frac{p_G}{\alpha}$. However if the chain comprises a cycle of length c, the only way to break it (if this local minimum is strong) is to wait for a simultaneous global exploration of all c sites, which happens as rarely as $\frac{p_G}{\alpha^c}$. Nevertheless, the appearance of such a cycle is possible only during the very few first steps, and the probability of such an event is as low as $p^c\alpha^c$ where p is the probability of one misplacement. Moreover, whenever a chain is broken, it cannot form again. Due to space limitation, some of the simple mathematical details were skipped.



(a) (b)

Figure 2: (a) Sample of 300 objects using the ellipse model after adding a Gaussian noise. (b) The detection result using the new MBC algorithm.

5.2 How the algorithm scales

A comparison now is essential to see how the proposed kernels affect the speed of convergence. The comparison will concern the different versions of the MBC algorithm. For the seek of completeness, we also add the MBD algorithm which is the fastest one in the stochastic family for this class of MPP models.[6] For details of the MBD algorithm, please refer to.[6,13] Other models for object counting exist, but we did not compare with those methods for two reasons: (1) the focus of this paper in on the optimization of MPP models, (2) methods such as[14] requires a training phase while our does not require.

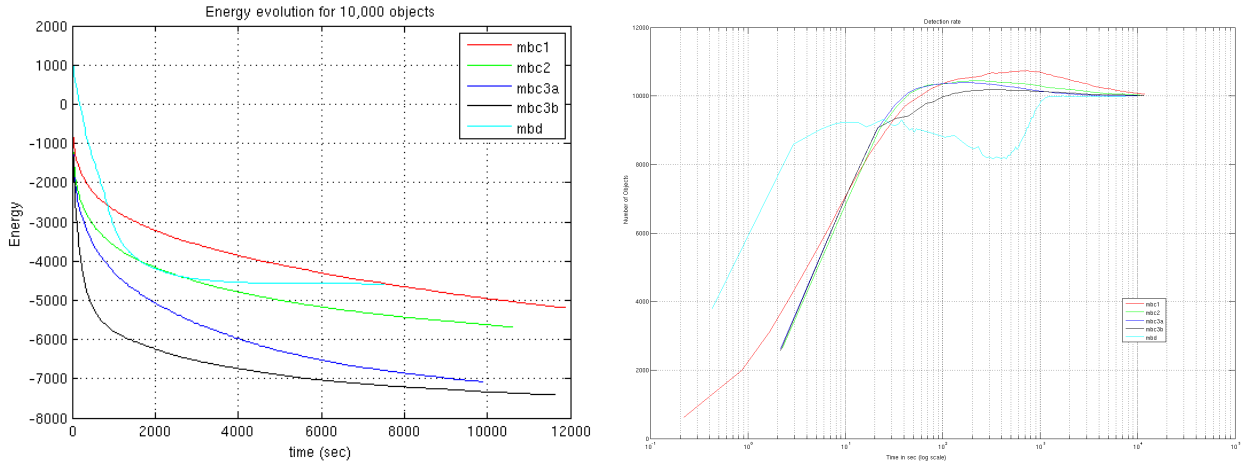
We tested the four algorithms on three samples containing respectively 300, 1000 and 10000 objects. These samples are generated by the same ellipse model introduced in.[7] The 300 object sample is shown in figure 2(a), and the detection using the new MBC algorithm is shown in figure 2(b). The comparison will present the evolution of the total energy of configurations versus time for each algorithm, as well as the object detection rate. In figure 3 we present the result on the largest sample of 10000 objects. In this figure we refere to the naive MBC by mbc1, mbc2 referes to MBC using belief propagation, mbc3a is our algorithm using only the new birth kernel and mbc3b refere to our algorithm using all kernels.

We use the following set of parameters: for the MBD algorithm, temperature $T = 1/50$, $\Delta T = 0.9997$, $\Delta\beta = 0.999$. For the MBC algorithm, the number of packets (objects for the naive MBC) is $R = 2000$, while for the first few iterations, we used $R = 3 \times R$ (for faster convergence), the packet size is 8 (no packets in the naive MBC).

For this configuration, figure 3(a) shows how the energy of configurations evolving with time for the four algorithms. The true value of the energy is $E(\omega) = -7740.95$. From the energy curves shown in figure 3(a,b), we can conclude the following:

- MBD converges faster that the naive MBC, but setting its parameters is not straightforward.
- Naive MBC (mbc1) will converge to the correct minimum, but it is slow.
- MBC using belief propagation (mbc2), is faster than the MBD and the naive MBC, but after a while its converges slowly.
- By adding only the new birth kernel, the algorithm (mbc3a) is much faster than the previous ones.
- By including all the proposed kernels, the algorithm (mbc3b) becomes very fast. Compared to the MBC using belief propagation, our algorithm is around **20 time faster**.

From the detection rate curve shown in figure 3(b), we conclude that:



(a) (b)

Figure 3: (a) Shows the energy evolution versus time on the 10000 objects' configuration. (b) Shows the number of objects detected as a function of the time spent (in log scale).

- All the algorithms find the correct number of objects in around 10000 sec.
- Our new algorithm using all kernels has the least variance to the true number of objects. In 300 sec, with a positive error of 2%, and then starts decreasing again to the correct value.
- The detection speed of the MBD algorithm is quite good after 1000 sec.

We can get a very good result from our new algorithm just after 100 sec for 10000 objects, while for the MBD we have to wait till 1000 sec.

5.3 Algorithm Parallelization

Since in this paper we consider only the first class of MPP models, happily it can be very easily parallelized. Given that the objects we are modeling, for *e.g.* flamingos, are independent, the resultant configuration from the optimization algorithm also should contain independent (non-overlapping) objects.

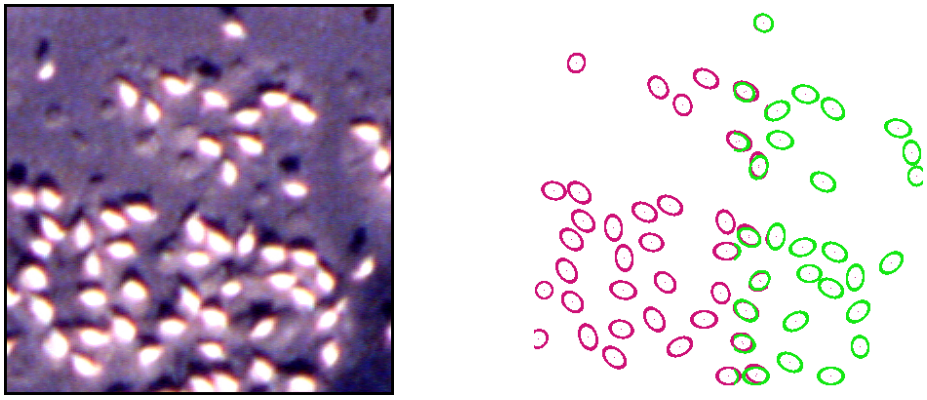
Given an input image, it should be *split* into n partitions, where n can be the number of cores in a processor, or number of machines used to solve the problem. Two issues to consider for splitting this image:

1. Each time we divide the image, there should be an overlap equal to the size of the largest existing object.
2. In many real cases, dividing the image into two equal halves at each split may not be the optimal choice for load balance. We propose to first calculate the birthmap, then make the split based on this map.

After splitting, each part is processed independently. The last step of the algorithm will be to *join* the result on the different parts. We propose to make the optimal join using graph cut, as it is done in every iteration in the Cuts step of the MBC algorithm.

To illustrate, consider the small sample of a flamingo colony shown in figure 4(a). This image is split vertically into two partitions, and each is optimized independently using the MBC algorithm. In figure 4(b), we overlap the detection result of the two partitions, one in pink and the other one in green. The common margin contains two detection result. In this example, nine flamingos are detected by both runs. We construct a graph, and to apply the same idea used in any of the MBC algorithms to select between two sets of configurations. This way of parallelization has the following advantages:

1. The cost of the fusion step is the of cost optimizing **one** small graph.



(a) (b)

Figure 4: (a) A small part of an aerial image of a flamingo colony. (b) Synthetic result on this sample, each color represent objects obtained independently on two cores.

2. This algorithm should scale linearly.
3. Since no communication is required while optimizing each partition, it can be done on different cores or different machines. The communication occur only twice: once to assign the task (split), and once to get the final result for fusion.
4. If an optimization algorithm other than MBC is needed, it can be used without any change, the fusion part will be the same.

6. APPLICATIONS

Recent technological advances in the development of airborne sensors have significantly increased the number of available satellites as well as their spatial and spectral resolutions. These advances, combined with the need for an effective environmental control, have favored the emergence of new remote sensing applications. Here we present results on three different applications.

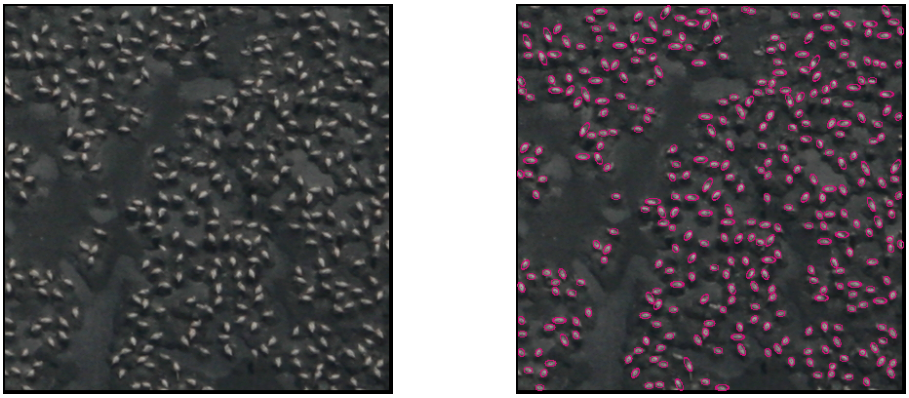
6.1 Ecological example 1: Flamingo counting

The unprecedented biotic crisis, due to human activities, follows from the destruction and fragmentation of natural habitats, overfishing, pollution and climate change. We consequently need long term monitoring systems allowing to estimate each year the number of breeding flamingos in the colonies and their breeding success. The development of an automatic counting system by photographic analysis of flamingo colonies could be a powerful and economic tool. We apply our MBC algorithm using the ellipse model developed in[13] to accomplish this automated counting. An aerial image of a small flamingo colony taken in Turkey is shown in figure 5.

6.2 Ecological example 2 : Tree counting

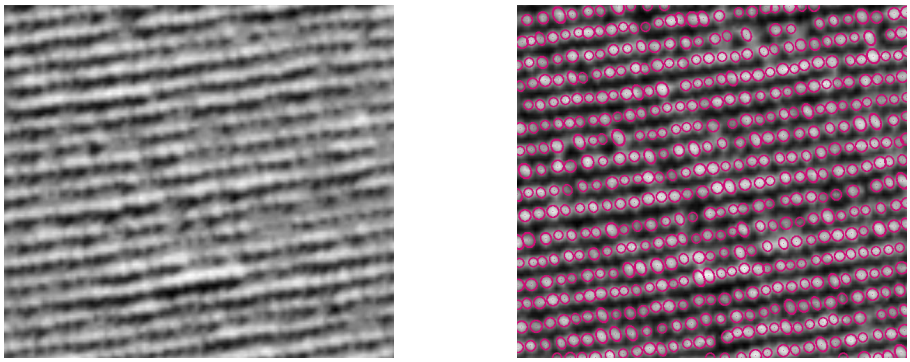
Brazilian eucalyptus plantations are among the most productive ecosystems in the world, providing wood sources for industry and energy. Monitoring these plantations on large surfaces is rather difficult and requires a lot of manpower. The aim is detecting individual tree crowns in localization and in size on large areas automatically, and provides important statistics for planting work.

In figure 6 we present a high resolution satellite image of eucalyptus plantation. We also considered the ellipse model while setting the acceptable overlapping coefficient to zero to obtain the result shown in figure 6(b).



(a) (b)

Figure 5: (a) An aerial image of a small flamingo colony taken in Turkey in 2006 (b) The detection result, each flamingo is surrounded by a pink ellipse.



(a) (b)

Figure 6: (a) Brazilian eucalyptus plantation (b) The detection result, each tree is surrounded by a pink ellipse.

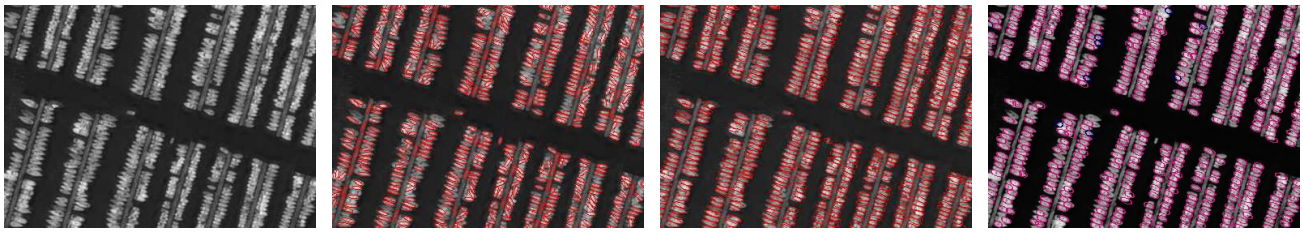
6.3 Surveillance: Boat detection

Here we present a more complicated example with a *peaky energy*. Given a satellite image of a port presented by figure 7(a), the aim is to detect and extract the boats. This energy is peaky because of objects' orientation: there exists a special structure (alignment) in the data. We use ellipses to approximate boats' shapes. In figure 7(b), we present the detection result, assuming uniform prior on the orientation of the ellipses $\theta \sim U_{[0,\pi[}$. The detection result is not that good, because of the complexity of finding this global minimum. One possible solution, is to run the algorithm for much longer time. Another solution will be to reduce the complexity of this problem by using a non-uniform prior on the orientation parameter. Another alternative would be to add an extra term to the prior term for alignment between objects. Using a non-uniform prior on the orientation parameter, we get the result shown in figure 7(c). Figure 7(d) shows the detection result by adding an alignment term to the prior energy.[15]

7. CONCLUSION AND FUTURE WORK

In this paper, we discussed the essential characteristics that should be considered to design efficient optimizers, while focusing on a selected class of models, namely Point Process Models. We presented the limitations of existing *birth-and-death* kernels, and, to overcome them, we introduced new kernels, both local and global.

In a theoretical analysis, we demonstrated the importance of both global and local perturbations. This analysis also showed how to select optimally the optimizer's parameters; in particular it quantified the optimal times spent in local and global explorations, in order to obtain the fastest convergence rates.



(a) (b) (c) (d)

Figure 7: (a) photograph of vessels in France ©CNES. (b) Detection result when using a uniform prior on boats orientations. (c) Detection result when using a non-uniform prior on the orientations. (d) Result when adding an alignment term to the prior energy.

We validated our algorithm on three different real applications, and showed its efficiency, as well as how it scales with the problem's size on synthetic data.

The MBC optimizer has proven to be simple and modular, while being very efficient. The proposed kernels are very simple and can be implemented easily. Moreover, adding new kernels to adapt the algorithm to specific applications, or even to tune the current algorithm, is possible without a lot of burden. While having all those simply designed kernels, the Cut step which is based on a binary graph cut, guarantees a global optimal selection.[2,7] Future work will consider extending this algorithm to solve more complex models, such as Point Process Models with sophisticated interactions.

REFERENCES

[1] Pham, M.-T., Gao, Y., Hoang, V., and Cham, T.-J., "Fast polygonal integration and its application in extending haar-like features to improve object detection," in [*IEEE Conf. on Computer Vision and Pattern Recognition*], 942 –949 (june 2010).

[2] Boykov, Y., Veksler, O., and Zabih, R., "Markov random fields with efficient approximations," in [*Conf. on Computer Vision and Pattern Recognition (CVPR)*], (1998).

[3] Ishikawa, H., "Exact optimization for Markov random fields with convex priors," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* **25** (2003).

[4] Geyer, C. J. and Møller, J., "Simulation Procedures and Likelihood Inference for Spatial Point Processes," *Scandinavian Journal of Statistics* **21**(4), 359–373 (1994).

[5] Green, P. J., "Reversible Jump Markov Chain Monte Carlo computation and Bayesian model determination," *Biometrika* **82**, 711–732 (1995).

[6] Descombes, X., Minlos, R., and Zhizhina, E., "Object extraction using a stochastic birth-and-death dynamics in continuum," *Journal of Mathematical Imaging and Vision* **33**(3), 347–359 (2009).

[7] Gamal Eldin, A., Descombes, X., and Zerubia, J., "Multiple birth and cut algorithm for point process optimization," in [*Proc. IEEE SITIS*], (December 2010).

[8] Gamal Eldin, A., Descombes, X., G., C., and Zerubia, J., "A fast multiple birth and cut algorithm using belief propagation," in [*Proc. IEEE International Conference on Image Processing (ICIP)*], (septembre 2011).

[9] Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. I., "An introduction to MCMC for machine learning," (september 2001).

[10] Andrieu, C. and Doucet, A., "Joint Bayesian model selection and estimation of noisy sinusoids via Reversible Jump MCMC," *IEEE Trans. on Signal Processing* **47**, 2667 –2676 (oct 1999).

[11] Perrin, G., Descombes, X., Zerubia, J., and Boureau, J., "Forest resource assessment using stochastic geometry," in [*Proc. Int. Precision Forestry Symposium*], (March 2006).

[12] Kesten, H., [*Percolation Theory for Mathematicians*], no. 2 in Progr. Prob. Statist., Birkhäuser, Mass. (1982).

[13] Descamps, S., Descombes, X., Béchet, A., and Zerubia, J., "Automatic flamingo detection using a multiple and death process," in [*ICASSP*], (March 2008).

[14] Lempitsky, V. S. and Zisserman, A., "Learning to count objects in images," in [*NIPS*], 1324–1332 (2010).
[15] Ben Hadj, S., Chatelain, F., Descombes, X., and Zerubia, J., "Parameter estimation for a marked point process within a framework of multidimensional shape extraction from remote sensing images," in [*Proc. ISPRS Tech. Com. III, Symposium on Photogrammetry Computer Vision and Image Analysis*], (September 2010).